KL-based Control of the Learning Schedule for Surrogate Black-Box Optimization

Ilya Loshchilov[1], Marc Schoenauer[2], and Michèle Sebag[3]

[1]LIS, EPFL, Switzerland. ilya.loshchilov@epfl.ch
[2]TAO, INRIA Saclay, France. marc.schoenauer@inria.fr
[3]CNRS, LRI UMR 8623, France. michele.sebag@lri.fr

Abstract

This paper investigates the control of an ML component within the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) devoted to black-box optimization. The known CMA-ES weakness is its sample complexity, the number of evaluations of the objective function needed to approximate the global optimum. This weakness is commonly addressed through surrogate optimization, learning an estimate of the objective function a.k.a. surrogate model, and replacing most evaluations of the true objective function with the (inexpensive) evaluation of the surrogate model. This paper presents a principled control of the learning schedule (when to relearn the surrogate model), based on the Kullback-Leibler divergence of the current search distribution and the training distribution of the former surrogate model. The experimental validation of the proposed approach shows significant performance gains on a comprehensive set of ill-conditioned benchmark problems, compared to the best state of the art including the quasi-Newton high-precision BFGS method.

Mots-clef: expensive black-box optimization, evolutionary algorithms, surogate models, Kullback-Leibler divergence, CMA-ES.

1 Introduction

As noted in [HCO12], the requirements on machine learning algorithms (ML) might be rather different depending on whether these algorithms are used in isolation, or as components in computational systems. Beyond the usual ML criteria of consistency and convergence speed, ML components should enforce some stability and controllability properties. Specifically, an ML component should *never* cause any catastrophic event (be it related to exceedingly high computational cost or exceedingly bad performance), over all system calls to this component. As a concrete example of controllability-enforcing strategy and contrarily to the ML usage, the stopping criterion of a Support Vector Machine algorithm should be defined in terms of number of quadratic programming iterations, rather than in terms of accuracy, since the convergence to the global optimum might be very slow under some circumstances [LS09]. In counterpart, such a strategy might reduce the predictive performance of the learned model in some cases, thus hindering the performance of the whole computational system. It thus becomes advisable that the ML component takes in charge the control of its hyper-parameters and even the schedule of its calls (when and how the predictive model should most appropriately be rebuilt). The automatic hyper-parameter tuning of an ML algorithm in the general case however shows to be a critical task, requiring sufficient empirical evidence.

This paper focuses on the embedding of a learning component within a distribution-based optimization algorithm [RK04, HMK03]. The visibility of such algorithms, particularly for industrial applications, is explained from their robustness w.r.t. (moderate) noise and multi-modality of the objective function [Han13], in contrast to classical optimization methods such as quasi-Newton methods (e.g. BFGS [Sha70]). One price to pay for this robustness is that the lack of any regularity assumption on the objective function leads to a large empirical sample complexity, i.e. number of evaluations of the objective function needed to approximate the global optima. Another drawback is the usually large number of hyper-parameters to be tuned for such algorithms to reach good performances. We shall however restrict ourselves in the remainder of the paper to the Covariance Matrix Adaptation Evolution

Strategy (CMA-ES) [HMK03], known as an almost parameterless distribution-based black-box optimization algorithm. This property is commonly attributed to CMA-ES invariance properties w.r.t. both monotonous transformations of the objective function, and linear transformations of the initial representation of the instance space (section 2.1).

The high sample complexity commonly prevents distribution-based optimization algorithms from being used on expensive optimization problems, where a single objective evaluation might require up to several hours (e.g. for optimal design in numerical engineering). The so-called surrogate optimization algorithms (see [Jin11] for a survey) address this limitation by coupling black-box optimization with learning of surrogate models, that is, local approximations of the objective function, and replacing most evaluations of the true objective function with the (inexpensive) evaluation of the surrogate function (section 2.2). The key issue of surrogate-based optimization is the control of the learning module (hyper-parameters tuning and update schedule).

In this paper, an integrated coupling of distribution-based optimization and rank-based learning algorithms, called KL-ACM-ES, is presented. The contribution compared to the state of the art [LSS12] is to analyze the learning schedule with respect to the drift of the sample distribution: Formally, after the surrogate model is trained from a given sample distribution, this distribution is iteratively modified along optimization. When to relearn the surrogate model depends on how fast the error rate of the surrogate model increases as the sample distribution moves away from the training distribution. Under mild assumptions, it is shown that the error rate increase can be bounded with respect to the Kullback-Leibler divergence between the training and the current distribution, yielding a principled learning schedule. The merit of the approach is empirically demonstrated as it shows significant performance gains on a comprehensive set of ill-conditioned benchmark problems [HFRA09a, HFRA09b], compared to the best state of the art including the quasi-Newton high-precision BFGS method.

The paper is organized as follows. For the sake of self-containedness, section 2 presents the Covariance Matrix Adaptation ES, and briefly reviews related work. Section 3 gives an overview of the proposed KL-ACM-ES algorithm and discusses the notion of drifting error rate. The experimental validation of the proposed approach is reported and discussed in section 4 and section 5 concludes the paper.

2 State of the art

This section summarizes Covariance Matrix Adaptation Evolution Strategy (CMA-ES), before discussing work related to surrogate-assisted optimization.

2.1 CMA-ES

Let f denote the objective function to be minimized on $\mathbb{R}^n$:

$$f : \mathbb{R}^n \mapsto \mathbb{R}$$

The so-called $(\mu/\mu_w, \lambda)$-CMA-ES [HMK03] maintains a Gaussian distribution on $\mathbb{R}^n$, iteratively used to generate λ samples, and updated based on the best (in the sense of f) μ samples out of the λ ones. Formally, samples $\boldsymbol{x}_{t+1}$ at time $t+1$ are drawn from the current Gaussian distribution P_{θ_t}, with $\theta_t = (\boldsymbol{m}_t, \sigma_t, \boldsymbol{C}_t)$:

$$\boldsymbol{x}_{t+1} \sim \mathcal{N}\big(\boldsymbol{m}_t, {\sigma_t}^2 \boldsymbol{C}_t\big) \tag{1}$$

where $\boldsymbol{m}_t \in \mathbb{R}^n$, $\sigma_t \in \mathbb{R}$, and $\boldsymbol{C}_t \in \mathbb{R}^{n\times n}$ respectively are the center of the Gaussian distribution (current best estimate of the optimum), the perturbation step size and the covariance matrix. The next distribution center $\boldsymbol{m}_{t+1}$ is set to the weighted sum of the best μ samples, denoting $\boldsymbol{x}_{t+1}^{(i:\lambda)}$ the i-th best sample out of the λ ones:

$$\boldsymbol{m}_{t+1} = \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{t+1}^{(i:\lambda)} \text{ with } \sum_{i=1}^{\mu} w_i = 1 \tag{2}$$

The next covariance matrix $\boldsymbol{C}_{t+1}$ is updated from $\boldsymbol{C}_t$ using both the local information about the search direction, given by $\frac{1}{\sigma_t}(\boldsymbol{x}_{t+1}^{(i:\lambda)} - \boldsymbol{m}_t)$, and the global information stored in the so-called evolution path $\boldsymbol{p}_{t+1}$ of the distribution center $\boldsymbol{m}$. For positive learning rates c_1 and c_μ ($c_1 + c_\mu \leq 1$) the update of the covariance matrix reads:

$$\begin{aligned}\boldsymbol{C}_{t+1} = (1 - c_1 - c_\mu)\boldsymbol{C}_t + c_1 \underbrace{\boldsymbol{p}_{t+1} \cdot \boldsymbol{p}_{t+1}{}^T}_{\text{rank-one update}} \\ + c_\mu \underbrace{\sum_{i=1}^{\mu} \frac{w_i}{{\sigma_t}^2} (\boldsymbol{x}_{t+1}^{(i:\lambda)} - \boldsymbol{m}_t) \cdot (\boldsymbol{x}_{t+1}^{(i:\lambda)} - \boldsymbol{m}_t)^T}_{\text{rank-}\mu\text{ update}}\end{aligned} \tag{3}$$

The step-size σ_{t+1} is likewise updated to best align the distribution of the actual evolution path of σ, and an evolution path under random selection.

As mentioned, the CMA-ES robustness and performances [HAR+10] are explained from its invariance properties. On the one hand, CMA-ES only considers the sample ranks after f; it thus does not

make any difference between optimizing f and $g \circ f$, for any strictly increasing scalar function g. On the other hand, the self-adaptation of the covariance matrix $\boldsymbol{C}$ makes CMA-ES invariant w.r.t. orthogonal transformations of the search space (rotation, symmetries, translation). Interestingly, CMA-ES can be interpreted in the Information-Geometric Optimization (IGO) framework [AAHO11]. IGO achieves a *natural gradient ascent* on the space of parametric distributions on the sample space X, using the Kullback-Leibler divergence as distance among distributions. It has been shown that the basic (μ, λ)-CMA-ES variant is a particular case of IGO when the parametric distribution space is that of Gaussian distributions [AAHO11].

2.2 Surrogate-based CMA-ES

Since the late 90s, many learning algorithms have been used within surrogate-based optimization, ranging from neural nets to Gaussian Processes (a.k.a. kriging) [USZ03, BSK05], using in particular the expected improvement [JSW98] as selection criterion. The surrogate CMA-ES algorithm most similar to our approach, ^{s}*ACM-ES [LSS12], interleaves two optimization procedures (Fig. 1):
The first one (noted CMA-ES #1) regards the optimization of the objective function f, assisted by the surrogate model $\widehat{f}$; the second one (CMA-ES #2) regards the optimization of the learning hyper-parameters α used to learn $\widehat{f}$.

More precisely, ^{s}*ACM-ES first launches CMA-ES on the true objective function f for a number of iterations n_{start}, and gathers the computed samples in a training set $\mathcal{E}_\theta = \{(x_i, f(x_i)), i = 1, \dots q\}$; the first surrogate model $\widehat{f}$ is learned from $\mathcal{E}_\theta$. ^{s}*ACM-ES then iterates the following process, referred to as epoch:
i) $\widehat{f}$ is optimized by CMA-ES for a given number of steps $\hat{n}$, leading from distribution P_θ to $P_{\theta'}$;
ii) CMA-ES is launched with distribution $P_{\theta'}$ on the true objective function, thereby building a new training set $\mathcal{E}_{\theta'}$;
iii) the previous and current training sets $\mathcal{E}_\theta$ and $\mathcal{E}_{\theta'}$ are used to adjust $\hat{n}$ and the other learning hyper-parameters α, and a new surrogate model $\widehat{f}$ is learned from $\mathcal{E}_{\theta'}$.

Surrogate model learning

Surrogate model $\widehat{f}$ is learned from the current training set $\mathcal{E}_\theta$, using Ranking-SVM [Joa05] together with the learning hyper-parameter vector α. For the sake of computational efficiency, a linear number of ranking constraints ($x_i \prec x_{i+1},\ i = 1 \dots q-1$) is used (assuming wlog that the samples in $\mathcal{E}_\theta$ are ordered by increasing value of f). By construction, the surrogate model thus is invariant under monotonous transformations of f.

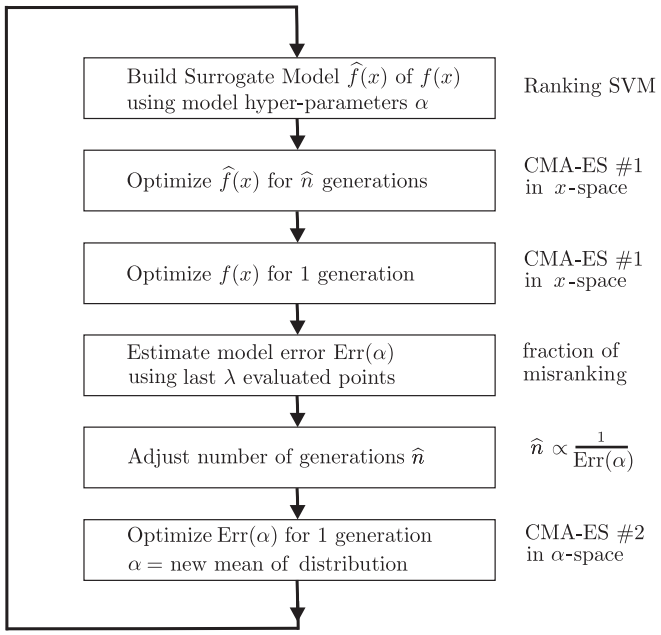

Figure 1: ^{s}*ACM-ES: interleaved optimization loop.

The invariance of $\widehat{f}$ w.r.t. orthogonal transformations of the search space is enforced by using a Radial Basis Function (RBF) kernel, which involves the inverse of the covariance matrix adapted by CMA-ES. Formally, the rank-based surrogate model is learned using the kernel K_C defined as: $K_C(\boldsymbol{x_i}, \boldsymbol{x_j}) = \mathrm{e}^{-\frac{(\boldsymbol{x_i}-\boldsymbol{x_j})^T \boldsymbol{C}^{-1} (\boldsymbol{x_i}-\boldsymbol{x_j})}{2s^2}}$, which corresponds to rescaling datasets $\mathcal{E}_\theta$ and $\mathcal{E}_{\theta'}$ using the transformation $\boldsymbol{x} \to \boldsymbol{C}^{-1/2}(\boldsymbol{x} - \boldsymbol{m})$, where $\boldsymbol{C}$ is the covariance matrix adapted by CMA-ES (and s is a learning hyper-parameter). Through this kernel, $\widehat{f}$ benefits from the CMA-ES efforts in identifying the local curvature of the optimization landscape.

Learning hyper-parameters

As mentioned, the control of the learning module (e.g. when to refresh the surrogate model; how many ranking constraints to use; which penalization weight) has a dramatic impact on the performance of the surrogate optimization algorithm. Both issues are settled in ^{s}*ACM-ES through exploiting the error of

the surrogate model $\widehat{f}$ trained from $\mathcal{E}_\theta$, measured on $\mathcal{E}_{\theta'}$. Formally, let $\ell(\widehat{f}, x, x')$ be 1 iff $\widehat{f}$ misranks x and x' compared to f, 1/2 if $f(x) = f(x')$ and 0 otherwise. The empirical ranking error of $\widehat{f}$ on $\mathcal{E}_{\theta'}$, referred to as empirical drift error rate, is defined as

$$\widehat{Err} = \frac{1}{|\mathcal{E}_{\theta'}|(|\mathcal{E}_{\theta'}|-1)} \sum_{\substack{x,x' \in \mathcal{E}_{\theta'} \\ x \neq x'}} \ell(\widehat{f}, x, x')$$

It is used to linearly adjust $\hat{n}$: close to 50%, it indicates that $\widehat{f}$ is no better than random guessing and that the surrogate model should have been rebuilt earlier ($\hat{n}$= 0); close to 0%, it inversely suggests that $\widehat{f}$ could have been used for a longer epoch ($\hat{n}$ is set to n_{max}, user-specified parameter of ^{s*}ACM-ES).

In the same spirit, $\mathcal{E}_\theta$ and $\mathcal{E}_{\theta'}$ are used to adjust the learning hyper-parameter vector, through a 1-iteration CMA-ES on the hyper-parameter space, minimizing the drift error rate of the surrogate model learned from $\mathcal{E}_\theta$ with hyper-parameters α'. The learning hyper-parameter vector α to be used in the next epoch is set to the center of the CMA-ES distribution on the hyper-parameter space.

2.3 Discussion

The main strength of ^{s*}ACM-ES is to achieve the simultaneous optimization of the objective function f together with the learning hyper-parameters (considering $\hat{n}$ as one among the learning hyper-parameters), thus only requiring the user to initially define their range of values and adjusting them online to minimize the drift error rate. It is worth noting that the automatic tuning of ML hyper-parameters is critical in general, particularly so when dealing with small sample sizes. The fact that the hyper-parameter tuning was found to be effective in the considered setting seems to be explained as the ML component in ^{s*}ACM-ES i) actually considers a sequence of learning problems defined by the successive distributions P_θ, ii) receives some feedback in each epoch about the α choice made in the previous epoch.

A significant weakness however is that the ^{s*}ACM-ES learning schedule is defined in terms of the number $\hat{n}$ of CMA-ES iterations in each epoch. However, the drift error rate should rather depend on how fast the optimization distribution P_θ is modified. This remark is at the core of the proposed algorithm.

3 Kullback-Leibler Divergence for Surrogate Model Control

The proposed KL-ACM-ES algorithm presented in this section differs from ^{s*}ACM-ES regarding the control of the learning schedule, that is, the decision of relearning the surrogate model. The proposed criterion is based on the Kullback-Leibler divergence between the distribution P_θ that was used to generate the training set of the current surrogate model $\widehat{f}$, referred to as training distribution, and the current working distribution $P_{\theta'}$ of CMA-ES. It is worth noting that since the Kullback-Leibler divergence depends only on P_θ and not on the parameterization of θ, this criterion is intrinsic and could be used for other distribution-based black-box optimization algorithms.

3.1 Analysis

Letting P_θ and $P_{\theta'}$ denote two distributions on the sample space, their Kullback-Leibler divergence noted $KL(P_{\theta'}||P_\theta)$ is defined as:

$$KL(P_{\theta'}||P_\theta) = \int_x \ln \frac{P_{\theta'}(x)}{P_\theta(x)} P_{\theta'}(dx). \tag{4}$$

Let $\widehat{f}$ denote in the following the surrogate model learned from $\mathcal{E}_\theta$, sampled after distribution P_θ. The idea is to retrain the surrogate model $\widehat{f}$ whenever it is estimated that its ranking error on $P_{\theta'}$ might be greater than a (user-specified) admissible error $Err_{admissible}$. Let us show that the difference between the generalization error of $\widehat{f}$ wrt P_θ and $P_{\theta'}$ is bounded depending on the KL divergence of P_θ and $P_{\theta'}$:

Proposition 1
The difference between the expectation of the ranking error of $\widehat{f}$ w.r.t. P_θ and w.r.t. $P_{\theta'}$, respectively noted $Err_{P_\theta}(\widehat{f})$ and $Err_{P_{\theta'}}(\widehat{f})$, is bounded by the square root of the KL divergence of P_θ and $P_{\theta'}$:

$$|Err_{P_\theta}(\widehat{f}) - Err_{P_{\theta'}}(\widehat{f})| \leq c_k \sqrt{KL(P_{\theta'}||P_\theta)} \tag{5}$$

with $c_k = 2\sqrt{2 \ln 2}$

Proof

$$\begin{aligned}
&|Err_{P_\theta}(\widehat{f}) - Err_{P_{\theta'}}(\widehat{f})| \\
&\quad = |\iint \ell(\widehat{f}, x, x')[P_\theta(x)P_\theta(x') - P_{\theta'}(x)P_{\theta'}(x')]dxdx'| \\
&\quad \leq |\iint (P_\theta(x)P_\theta(x') - P_{\theta'}(x)P_{\theta'}(x'))dxdx'| \\
&\quad \leq 2||P_\theta - P_{\theta'}||_1 \\
&\quad \leq c_k \sqrt{KL(P_{\theta'}||P_\theta)}
\end{aligned}$$

where the first inequality follows from the fact that ℓ is positive and bounded by 1, and the second from

applying the usual trick $(ab - cd) = a(b - d) + d(a - c)$, and separately integrating w.r.t. x and x'. The last inequality follows from the result that for any two distributions P and Q,

$$KL(P||Q) \geq \frac{1}{2 \ln 2} ||P - Q||_1^2$$

See, e.g., Cover and Thomas (1991, Lemma 12.6.1, pp. 300–301). □

The second step, bounding the difference between the empirical and the generalization ranking error of $\widehat{f}_\theta$, follows from the statistical learning theory applied to ranking [CLV08, AN09, Rej12].

Let us first recall the property of *uniform loss stability*. A ranking algorithm has the uniform loss stability property iff for any two q-size samples $\mathcal{E}$ and $\mathcal{E}'$ drawn after the same distribution and differing by a single sample, if $\widehat{f}$ and $\hat{f}'$ are the ranking functions learned from respectively $\mathcal{E}$ and $\mathcal{E}'$, the following holds for any (x, x') pair, for some β_q that only depends on q:

$$|\ell(\widehat{f}, x, x') - \ell(\hat{f}', x, x')| < \beta_q$$

Proposition 2 [AN09]
If the ranking algorithm has the uniform loss stability property, then for any $0 < \delta < 1$, with probability at least $1-\delta$ (over the draw of the q-size test set $\mathcal{E}_\theta$ drawn according to P_θ), the generalization ranking error of $\widehat{f}$ is bounded by its empirical error on $\mathcal{E}_\theta$, plus a term that only depends on q:

$$|Err_{P_\theta}(\widehat{f}) - \widehat{Err_{P_\theta,q}}(\widehat{f})| < 2\beta_q + (q\beta_q + 1)\sqrt{\frac{2 \ln 1/\delta}{q}} \tag{6}$$

From Eqs. (5) and (6), it is straightforward to show that with probability at least $1-\delta/2$ the empirical error of $\widehat{f}$ on $\mathcal{E}_{\theta'}$ is bounded by a term that only depends on q, plus the square root of the KL divergence between P_θ and $P_{\theta'}$:

$$\begin{aligned} \widehat{Err_{P_{\theta'},q}}(\widehat{f}) &< 4\beta_q + 2(q\beta_q + 1)\sqrt{\tfrac{2 \ln 1/\delta}{q}} \\ &+ \widehat{Err_{P_\theta,q}}(\widehat{f}) + c_k\sqrt{KL(P_\theta||P_{\theta'})} \end{aligned} \tag{7}$$

In the context of this work, Ranking-SVM have the uniform loss stability property [AN09]. Hence, if the q-dependent terms of Eq. (7) are small enough, it is possible, given a parameter $Err_{admissible}$, to define a threshold KL_{thresh} such that, provided that the KL divergence between P_θ and $P_{\theta'}$ remains below KL_{thresh}, the empirical error of $\widehat{f}_\theta$ on $\mathcal{E}_{\theta'}$ remains bounded by $Err_{admissible}$ with high probability. In practice however, as noted by [CLV08], it is well known that the above bound is often quite loose, even in the case where ℓ is convex and σ-admissible [AN09]. For instance, using the hinge loss function proposed in [AN09], the constant β_q of Eq. (7) is inversely proportional to q, and the q-dependent terms of Eq. (7) vanish for large q. However, as discussed in section 1, the context of application of ML algorithms is driven here by the optimization goal. In particular, the number of training samples q has to be kept as small as possible. This is why an empirical alternative for the update of KL_{thresh} has been investigated, inspired by the trust-region paradigm from classical optimization.

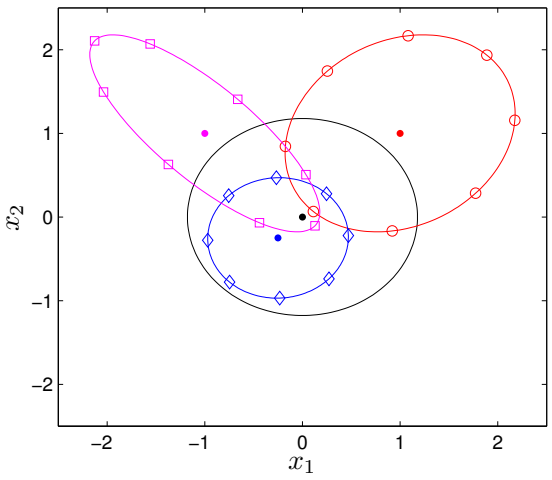

Figure 2: Ellipsoidal 95% confidence regions of the Gaussian distribution P_θ (black thin line) and three other Gaussian distributions $P_{\theta'}$ (color marked lines) with same $KL(P_{\theta'}||P_\theta)$.

3.2 Adaptive adjustment of KL_{thresh}

Let us remind that, for P_θ and $P_{\theta'}$ (respectively defined by $\theta = (\boldsymbol{m}, \boldsymbol{C})$ and $\theta' = (\boldsymbol{m}', \boldsymbol{C}')$) the Kullback-Leibler divergence of P_θ and $P_{\theta'}$ has a closed form expression,

$$\begin{aligned} KL(P_{\theta'}||P_\theta) = \frac{1}{2}[&tr(\boldsymbol{C}^{-1}\boldsymbol{C}') + \\ &(\boldsymbol{m} - \boldsymbol{m}')^T \boldsymbol{C}^{-1}(\boldsymbol{m} - \boldsymbol{m}') - n - \ln(\tfrac{det\boldsymbol{C}'}{det\boldsymbol{C}})]. \end{aligned} \tag{8}$$

Note that increasing values of $KL(P_{\theta'}||P_\theta)$ might reflect different phenomenons; they can be due to either differences in the distribution center or in the scaling of the covariance matrix (see the examples on Fig. 2).

The sought threshold KL_{thresh} finally is interpreted in terms of trust regions [MS83]. Classical heuristic optimization methods often proceed by associating to a region of the search space the (usually) quadratic surrogate model approximating the objective function in this region. Such a region, referred to as trust region, is assessed from the ratio of expected improvement measured on the surrogate model, and the improvement on the true objective. Depending on this ratio, the trust region is expanded or restricted.

The proposed KL-based control of the learning schedule can be viewed as a principled way to adaptively control the dynamics of the trust region, with three differences. Firstly, the trust region is here defined in terms of distributions on the search space: the trust region is defined as the set of all distributions $P_{\theta'}$ such that $KL(P_{\theta'}||P_\theta) < KL_{thresh}$. Secondly, this trust region is assessed a posteriori from the empirical error of the surrogate model, on the first distribution $P_{\theta'}$ *outside* the trust region. Thirdly, this assessment is exploited to adjust the KL "radius" of the next trust region, KL_{thresh}.

Finally, KL_{thresh} is set such that $\log KL_{thresh}$ is inversely proportional to the relaxed surrogate error Err:

$$\ln(KL_{thresh}) \leftarrow \frac{\tau_{err} - Err}{\tau_{err}} \ln(KL_{Max}), \qquad (9)$$

where τ_{err} is an error threshold (experimentally set to .45), KL_{Max} is the maximal allowed KL divergence when $\widehat{f}$ is ideal, and the relaxed surrogate error Err is computed as $Err = (1-\alpha)Err + \alpha \widehat{Err}_{P_{\theta',q}}(\widehat{f})$, in order to moderate the effects of $\widehat{Err}_{P_{\theta',q}}(\widehat{f})$ high variance. The log-scaling is chosen on the basis of preliminary experiments; its adjustment online is left for further work.

Finally, KL-ACM-ES differs from ^{s}*ACM-ES in two points:

- The second step of the algorithm (Fig. 1) thus becomes "Optimize $\widehat{f}$ while $KL(P_{\theta'}||P_\theta) \leq KL_{thresh}$;
- The fifth step of the algorithm (Fig. 1) becomes "Adjust KL_{thresh}" from Eq. (9).

4 Experimental validation

This section presents the experimental validation of the proposed KL-ACM-ES algorithm compared to various CMA-ES algorithms including ^{s}*ACM-ES, and the quasi Newton BFGS algorithm [Sha70] on the BBOB noiseless and noisy benchmark suite [HAFR12].

4.1 Experimental Setting

For reproducibility, the Matlab source code of KL-ACM-ES is made available together with its default parameters[1]. After preliminary experiments, $\ln(KL_{Max})$ is set to 6; the CMA-ES λ parameter used when optimizing $\widehat{f}_\theta$ is multiplied by 100 when the empirical error rate of $\widehat{f}_\theta$ on $\mathcal{E}_{\theta'}$ is less than .35. The comparative validation firstly involves ^{s}*ACM-ES with its default parameters[2]. The CMA-ES variant used within ^{s}*ACM-ES and KL-ACM-ES is the state of the art BIPOP-active CMA-ES algorithm [HR10]. The comparative validation also involves the Quasi-Newton BFGS method. Indeed, BFGS suffers from known numerical problems on ill-conditioned problems (see [Pow87] for an extensive discussion). This limitation is overcome by considering instead the 32-decimal digit precision arithmetic version of BFGS, referred to as pBFGS[3] and included in the high-precision arithmetic package ARPREC [BYLT02].

The algorithms have been compared on the twenty-four 20-dimensional noiseless and thirty noisy benchmark problems of the BBOB suite [HFRA09a, HFRA09b] with different known characteristics: separable, non-separable, unimodal, multi-modal, ill-conditioned, deceptive, functions with and without weak global structure. For each problem, 50 uniformly generated orthogonal transformations of f are considered.

For each problem and each algorithm, 15 independent runs (each one with a randomly chosen position for the optimum) are launched. The performance of each algorithm is reported as the median optimum value (in log scale) vs the number of evaluations of f (Fig. 3), or as the empirical cumulative distribution of success for solving sets of similar functions (Fig. 4, see caption there). All algorithms are initialized with samples uniformly drawn in $[-5, 5]^{20}$.

4.2 Case study: the Rosenbrock function

A first study is conducted on the 20-dimensional Rosenbrock function and its second and fourth power. Rosenbrock function is defined as:

$$f_{Ros}(x) = \sum_{i=1}^{n-1} \left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$$

[1] https://sites.google.com/site/acmesKL/

[2] https://sites.google.com/site/acmesgecco/

[3] https://sites.google.com/site/highprecisionbfgs/.
For gradient approximations by finite differences $\epsilon = 10^{-20}$ is used in pBFGS instead of $\epsilon = 10^{-8}$ in BFGS.

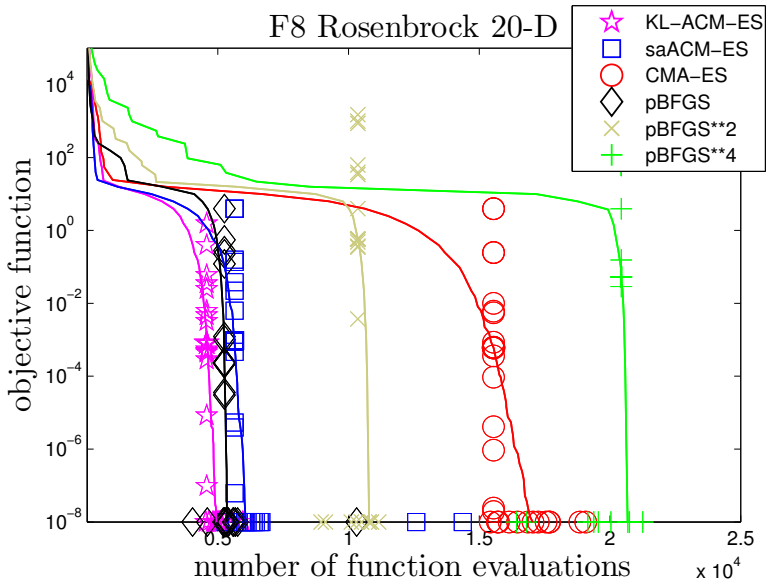

Figure 3: Comparative performance of KL-ACM-ES compared to high-precision BFGS and CMA-ES variants (see text) on the 20-dimensional Rosenbrock function f_{Ros}, f_{Ros}^2 and f_{Ros}^4. The medium number of function evaluations (out of 15 runs) to reach the target objective value 10^{-8} was computed and the corresponding run is shown. Markers show the objective value reached in each run after a given number of function evaluations.

The empirical results (Fig. 3) show that both surrogate-based optimization algorithms ^{s}*ACM-ES and KL-ACM-ES improve on the CMA-ES variant baseline by a factor ranging from 2.5 to 3.

KL-ACM-ES outperforms ^{s}*ACM-ES and pBFGS on the Rosenbrock function (BFGS, omitted for clarity, behaves like pBFGS as the Rosenbrock function is not ill-conditioned). The merits of the invariance w.r.t. monotonous transformations of the objective function shine as the pBFGS behavior is significantly degraded on f_{Ros}^2 (legend pBFGS**2) and f_{Ros}^4 (legend pBFGS**4) compared to f_{Ros}, slowing down the convergence by a factor of about 4 for f_{Ros}^4. Quite the contrary, all CMA-ES variants including ^{s}*ACM-ES and KL-ACM-ES behave exactly the same on all three functions, by construction.

The performance improvement of KL-ACM-ES on ^{s}*ACM-ES and pBFGS is over 20%.

4.3 Results on BBOB problems

Fig. 4 displays the overall results on BBOB benchmark. Similar functions are grouped, thus distinguishing the cases of separable, moderately difficult, ill-conditioned, multi-modal, weakly structured multi-modal objective functions (last plot aggregates all functions). The *Best 2009* reference corresponds to the (virtual) best performance reached over the portfolio of all algorithms participating in the BBOB 2009 contest (portfolio oracle; note that the high-precision pBFGS was not included in the portfolio).

On separable objective functions, pBFGS dominates all CMA-ES variants for small numbers of evaluations. Then KL-ACM-ES catches up, followed by ^{s}*ACM-ES and the other CMA-ES variants. On moderate and ill-conditioned functions, KL-ACM-ES dominates all other algorithms and it even improves over *Best 2009*. On multi-modal and weakly structured multi-modal functions, KL-ACM-ES is dominated by ^{s}*ACM-ES; in the meanwhile BFGS and pBFGS alike yield poor performance.

Finally, on the overall plot, KL-ACM-ES shows good performances compared to the state of the art, prodding the *Best 2009* curve in the median region due to the significant progress made on ill-conditioned functions.

Besides the merits of rank-based optimization, these results demonstrate the relevance of using high-precision BFGS instead of BFGS for solving ill-conditioned optimization problems. A caveat is that high-precision computations require the source code to be rewritten (not only in the optimization algorithm,

but also in the objective function), which is hardly feasible in standard black-box scenarios − even when the objective source code is available.
Albeit pBFGS significantly dominates BFGS, its behavior is shown to be significantly degraded when the scaling of the objective function differs from the "desirable" quadratic BFGS scaling.

5 Discussion and future work

This paper investigates the control of an ML component within a compound computational system. In order to provide a steady and robust contribution to the whole system, it is emphasized that an ML component must take in charge the adjustment of its learning hyper-parameters, and also *when and how frequently this ML component should be called*. In the meanwhile and as could have been expected, such an autonomous ML component must accommodate applicative priorities and experimental conditions which differ from those commonly faced by ML algorithms in isolation.

A first attempt toward building such an autonomous ML component in the context of black-box distribution-based optimization has been presented. In this context, the ML component is meant to supply an estimate of the objective optimization function; it faces a sequence of learning problems, drawn from moving distributions on the instance space. The first contribution is to show how the KL divergence between successive distributions can yield a principled control of the learning schedule, that is, the decision to relearn an estimate of the objective optimization function. The second contribution is that the KL-ACM-ES algorithm implementing this learning schedule control improves on the best state-of-the-art distribution-based optimization algorithms and quasi-Newton methods, on a comprehensive suite of ill-conditioned benchmark problems.

Further work will examine how to further enhance the autonomy of the ML component, e.g. when facing multi-modal objective functions. Alternative comparison-based surrogate models will also be considered, such as Gaussian Processes for ordinal regression [CG05]. Finally, as shown by [HK12], quasi-Newton methods can be interpreted as approximations of Bayesian linear regression under varying prior assumptions; a prospective research direction is to replace the linear regression by ordinal regression-based Ranking SVM or Gaussian Processes in order to derive a version of BFGS invariant w.r.t. monotonous transformations of the objective function f.

References

[AAHO11] L. Arnold, A. Auger, N. Hansen, and Y. Ollivier. Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles. *ArXiv e-prints*, June 2011.

[AN09] S. Agarwal and P. Niyogi. Generalization Bounds for Ranking Algorithms via Algorithmic Stability. *J. of Machine Learning Research*, 10:441–474, 2009.

[BSK05] D. Buche, N. N. Schraudolph, and P. Koumoutsakos. Accelerating Evolutionary Algorithms with Gaussian Process Fitness Fn Models. *IEEE Trans. Systems, Man and Cybernetics*, 35(2):183–194, 2005.

[BYLT02] D.H. Bailey, H. Yozo, X.S. Li, and B. Thompson. ARPREC: An Arbitrary Precision Computation Package. Technical report, Ernest Orlando Lawrence Berkeley National Laboratory, CA, 2002.

[CG05] W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In *Proc. 22nd ICML*, pages 137–144. ACM, 2005.

[CLV08] S. Clemençon, G. Lugosi, and N. Vayatis. Ranking and Empirical Minimization of U-statistics. *The Annals of Statistics*, 36(2):844–874, 2008.

[HAFR12] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-Parameter Black-Box Optimization Benchmarking 2012: Experimental Setup. Technical report, INRIA, 2012.

[Han13] N. Hansen. References to CMA-ES Applications. Website, January 2013. Available online at http://www.lri.fr/ hansen/cmaapplications.pdf.

[HAR+10] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In *GECCO Companion*, pages 1689–1696, 2010.

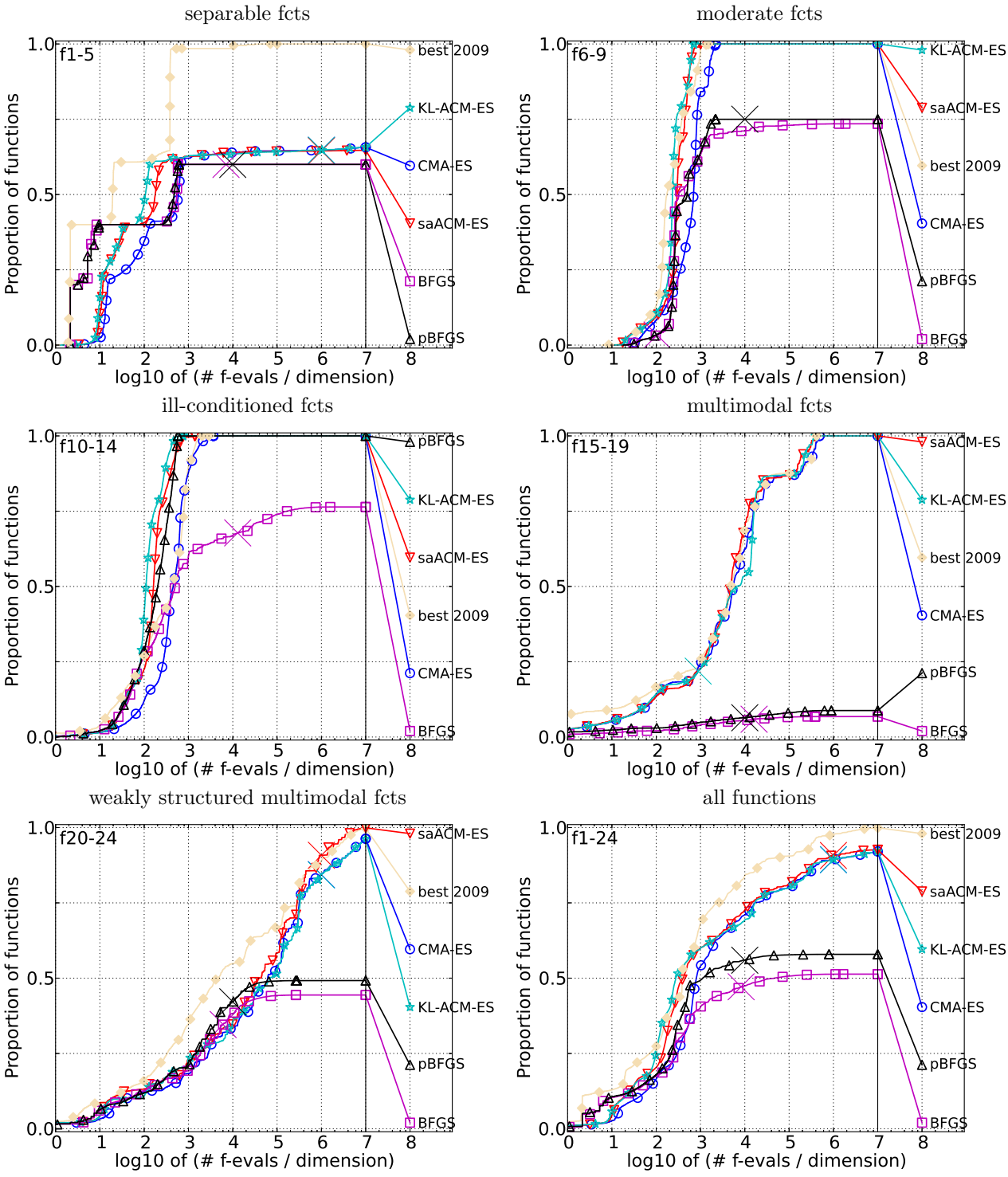

Figure 4: Empirical cumulative distributions (i.e. proportion of functions solved – up to given target precisions in $10^{[-8..2]}$) of the number of objective function evaluations divided by dimension for all functions (last plot) and subgroups of similar functions (other plots) in 20-D. The "best 2009" line indicates the BBOB 2009 "portfolio oracle", the aggregation of the best results for each function. The proposed algorithm is depicted as KL-ACM-ES. A detailed description of these representations of BBOB results can be found in [HAR+10].

[HCO12] P. Hennig, J. P. Cunningham, and M. A. Osborne, editors. *Probabilistic Numerics Workshop, NIPS*, 2012.

[HFRA09a] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA, 2009.

[HFRA09b] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-Parameter Black-Box Optimization Benchmarking 2009: Noisy Functions Definitions. Research Report RR-6869, INRIA, 2009.

[HK12] P. Hennig and M. Kiefel. Quasi-Newton Methods: A New Direction. *Preprint arXiv:1206.4602*, 2012.

[HMK03] N. Hansen, S.D. Müller, and P. Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation. *Evolutionary Computation*, 11(1):1–18, 2003.

[HR10] N. Hansen and R. Ros. Benchmarking a weighted negative covariance matrix update on the BBOB-2010 noiseless testbed. In *GECCO Companion*, pages 1673–1680, New York, NY, USA, 2010. ACM.

[Jin11] Y. Jin. Surrogate-Assisted Evolutionary Computation: Recent Advances and Future Challenges. *Swarm and Evolutionary Computation*, pages 61–70, 2011.

[Joa05] T. Joachims. A Support Vector Method for Multivariate Performance Measures. In *Proc. 22nd ICML*, pages 377–384. ACM, 2005.

[JSW98] Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *J. of Global Optimization*, 13(4):455–492, December 1998.

[LS09] N. List and H.U. Simon. SVM-optimization and steepest-descent line search. In *Proceedings of the 22nd COLT*, 2009.

[LSS12] I. Loshchilov, M. Schoenauer, and M. Sebag. Self-Adaptive Surrogate-Assisted CMA-ES. In T. Soule and J.H. Moore, editors, *Proc. GECCO*, pages 321–328. ACM Press, July 2012.

[MS83] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

[Pow87] M.J.D. Powell. Updating Conjugate Directions by the BFGS Formula. *Mathematical Programming*, 38(1):29–46, 1987.

[Rej12] Wojciech Rejchel. On ranking and generalization bounds. *The Journal of Machine Learning Research*, 98888:1373–1392, 2012.

[RK04] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

[Sha70] D. F. Shanno. Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

[USZ03] H. Ulmer, F. Streichert, and A. Zell. Evolution Strategies assisted by Gaussian Processes with Improved Pre-Selection Criterion. In *IEEE Congress on Evolutionary Computation*, pages 692–699, 2003.